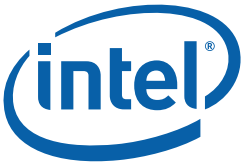


Intel® Pentium® M Processor on 90 nm Process with 2-MB L2 Cache

Specification Update

February 2007

Document Number: 302209-021

Contents

Revision History

Revision	Description	Date
-001	• Initial Release	May 2004
-002	• Updated Processor Identification Table	June 2004
-003	• Updated Processor Identification Table • Added Erratum X10	July 2004
-004	• Added Erratum X11, X12 and X13	October 2004
-005	• Added Erratum X14, X15 and X16	November 2004
-006	• Added Erratum X17 and X18	December 2004
-007	• Updated Processor Identification Table: Added C-0 S-Specs • Updated Summary Tables of Changes • Added Erratum X19 – X21	February 2005
-008	• Updated Summary of Tables of Changes • Updated Processor Identification Table • Added Specification Clarification X1	April 2005
-009	• Updated Processor Identification Table: Shading corrected from revision -008.	April 2005
-010	• Added Erratum X22 • Added Specification Clarification X2	May 2005
-011	• Updated Summary Tables of Changes • Added Erratum X23, X24 and X25	June 2005
-012	• Updated Processor Identification Table 1	July 2005
-013	• Updated Affected and Related Documents Tables • Updated Processor Identification Table 1 • Removed Erratum X13 (which was duplicate of X1)	July 2005
-014	• Added Specification Change X1	October 2005
-015	• Added Erratum X26	November 2005
-016	• Added Erratum X27 and X28	December 2005
-017	• Added Erratum X29 - X35 • Removed Specification Clarification X1; refer to Section 18.8 of the *IA-32 Intel® Architecture Software Developer's Manual, Volume 3B* for detailed Time Stamp Counter information	January 2006
-018	• Added Errata X36 and X37 • Updated Erratum X35 • Updated Processor Identification Table 1	April 2006

Revision	Description	Date
-019	• Added Errata X38-X47 • Updated Errata X23 and X35 • Updated Related Documents Table • Update CPU Identification table with new processors • Added new processors to the key in the Summary Tables of Changes	November 2006
-020	• Added Errata X47-X50 • Updated CPU Identification table with new processors	December 2006
-021	• Added Errata X51, X52	February 2007

Preface

This document is an update to the specifications contained in the documents listed in the following Affected Related Documents table. It is a compilation of device and document errata and specification clarifications and changes, and is intended for hardware system manufacturers and for software developers of applications, operating system, and tools.

Information types defined in the Nomenclature section of this document are consolidated into this update document and are no longer published in other documents. This document may also contain information that has not been previously published.

Affected Documents

Document Title	Document Number
The Intel® Pentium® M Processor on 90 nm Process with 2-MB L2 Cache Datasheet	302189-007
The Intel® Pentium® M Processor with 2-MB L2 Cache and 533-MHz Front Side Bus Datasheet	305262-002

Related Documents

Document Title	Document Number
Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 1	253665
Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 2A	253666
Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 2B	253667
Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3A	253668
Intel® 64 and IA-32 Architectures Software Developer's Manual, Volume 3B	253669
IA-32 Intel® Architecture Optimization Reference Manual	248966

Nomenclature

S-Spec Number is a five-digit code used to identify products. Products are differentiated by their unique characteristics, e.g., core speed, L2 cache size, package type, etc. as described in the processor identification information table. Care should be taken to read all notes associated with each S-Spec number.

Errata are design defects or errors. Errata may cause the behavior of the Intel® Pentium® M processor on 90 nm process with 2-MB L2 cache, to deviate from published specifications. Hardware and software, designed to be used with any given processor, must assume that all errata documented for that processor are present on all devices unless otherwise noted.

Documentation Changes include typos, errors, or omissions from the current published specifications. These changes will be incorporated in the next release of the specifications.

Specification Clarifications describe a specification in greater detail or further highlight a specification's impact to a complex design situation. These clarifications will be incorporated in the next release of the specifications.

Specification Changes are modifications to the current published specifications for the Pentium M processor. These changes will be incorporated in the next release of the specifications.

Summary Tables of Changes

The following table indicates the Specification Changes, Errata, Specification Clarifications or Documentation Changes, which apply to the listed MCH steppings. Intel intends to fix some of the errata in a future stepping of the component, and to account for the other outstanding issues through documentation or Specification Changes as noted. This table uses the following notations:

Codes Used in Summary Table

Stepping

X:	Erratum, Specification Change or Clarification that applies to this stepping.
(No mark) or (Blank Box):	This erratum is fixed in listed stepping or specification change does not apply to listed stepping.

Status

Doc:	Document change or update that will be implemented.
PlanFix:	This erratum may be fixed in a future stepping of the product.
Fixed:	This erratum has been previously fixed.
NoFix:	There are no plans to fix this erratum.

Row

Shaded:	This item is either new or modified from the previous version of the document.

Each specification update item is prefixed with a capital letter to distinguish the product. The key below details the letters that are used in Intel's microprocessor Specification Updates:

A = Dual-Core Intel® Xeon® processor 7000$^{\Delta}$ sequence

B = Mobile Intel® Pentium® II processor

C = Intel® Celeron® processor

D = Intel® Pentium® II Xeon® processor

E = Intel® Pentium® III processor

F = Intel® Pentium® processor Extreme Edition and Intel® Pentium® D processor

G = Intel® Pentium® III Xeon® processor

H = Mobile Intel® Celeron® processor at 466 MHz, 433 MHz, 400 MHz, 366 MHz, 333 MHz, 300 MHz, and 266 MHz

J = 64-bit Intel® Xeon® processor MP with 1-MB L2 cache

K = Mobile Intel® Pentium® III Processor – M

L = Intel® Celeron® D processor

M =Mobile Intel® Celeron® processor

N = Intel® Pentium® 4 processor

O = Intel® Xeon® processor MP

P = Intel® Xeon® processor

Q = Mobile Intel® Pentium® 4 processor supporting Hyper-Threading Technology† on 90 nm technology process

R = Intel® Pentium® 4 processor on 90 nm process

S = 64-bit Intel® Xeon® processor with 800 MHz system bus (1-MB and 2-MB L2 cache versions)

T = Mobile Intel® Pentium® 4 processor – M

U = 64-bit Intel® Xeon® processor MP with up to 8-MB L3 cache

V = Mobile Intel® Celeron® processor on .13 Micron process in Micro-FCPGA package

W = Intel® Celeron® M processor

X = Intel® Pentium® M processor on 90 nm process with 2-MB L2 cache

Y = Intel® Pentium® M processor

Z = Mobile Intel® Pentium® 4 Processor with 533 MHz System Bus

AA = Intel® Pentium® processor Extreme Edition and Intel® Pentium® D processor on 65 nm process

AB = Intel® Pentium® 4 processor on 65 nm process

AC = Intel® Celeron® Processor in 478 Pin Package

AE = Intel® Core™ Duo processor and Intel® Core™ Solo processor on 65 nm process

AF = Dual-Core Intel® Xeon® processor LV

AG = Dual-Core Intel® Xeon® Processor 5100$^{\Delta}$ Series

AH = Intel® Core™2 Duo mobile processor

AI = Intel® Core™2 Extreme Processor X6800$^{\Delta}$ and Intel® Core™2 Duo Desktop Processor E6000$^{\Delta}$ Sequence

AJ = Quad-Core Intel® Xeon® Processor 5300 Series

AK = Intel® Core™2 Extreme quad-core processor QX6700Δ and Intel® Core™2 Quad processor Q6600Δ

AL = Dual-Core Intel® Xeon® Processor 7100$^{\Delta}$ Series

AO = Quad-Core Intel® Xeon® Processor 3200 Series

AP = Dual-Core Intel® Xeon™ Processor 3000 Series

Δ Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. See http://www.intel.com/products/processor_number for details

Note: The specification updates for the Pentium processor, Pentium Pro processor, and other Intel products do not use this convention.

NO.	B1	C0	Plans	ERRATA
X1	X	X	No Fix	Code Segment (CS) Is Wrong on SMM Handler when SMBASE Is Not Aligned
X2	X	X	No Fix	IFU/BSU Deadlock May Cause System Hang
X3	X	X	No Fix	Memory Aliasing with Inconsistent A and D Bits May Cause Processor Deadlock
X4	X	X	No Fix	RDMSR or WRMSR to Invalid MSR Address May Not Cause GP Fault
X5	X	X	No Fix	Unable To Disable Reads/Writes to Performance Monitoring Related MSRs
X6	X	X	No Fix	Move to Control Register Instruction May Generate a Breakpoint Report
X7	X	X	No Fix	Error in Instruction Fetch Unit (IFU) Can Result in an Erroneous Machine Check-Exception (#MC)
X8	X	X	No Fix	Code Fetch Matching Disabled Debug Register May Cause Debug Exception
X9	X	X	No Fix	Upper Four PAT Entries Not Usable with Mode B or Mode C Paging
X10	X	X	No Fix	SSE/SSE2 Streaming Store Resulting in a Self-Modifying Code (SMC) Event May Cause Unexpected Behavior
X11	X	X	No Fix	Code Segment Limit Violation May Occur on 4 Gigabyte Limit Check
X12	X	X	No Fix	FST Instruction with Numeric and Null Segment Exceptions may cause General Protection Faults to be Missed and FP Linear Address (FLA) Mismatch
X13				Removed, see Erratum X1.
X14	X	X	No Fix	Page with PAT (Page Attribute Table) Set to USWC (Uncacheable Speculative Write Combine) While Associated MTRR (Memory Type Range Register) is UC (Uncacheable) May Consolidate to UC
X15	X	X	No Fix	Under Certain Conditions LTR (Load Task Register) Instruction May Result in System Hang
X16	X	X	No Fix	Loading from Memory Type USWC (Uncacheable Speculative Write Combine) May Get Its Data Internally Forwarded from a Previous Pending Store
X17	X	X	No Fix	FXSAVE after FNINIT without an Intervening FP (Floating Point) Instruction May Save Uninitialized Values for FDP (x87 FPU Instruction Operand (Data) Pointer Offset) and FDS (x87 FPU Instruction Operand (Data) Pointer Selector)
X18	X	X	No Fix	FSTP (Floating Point Store) Instruction under Certain Conditions May Result In Erroneously Setting a Valid Bit on an FP (Floating Point) Stack Register
X19		X	No Fix	An Execute Disable Bit Violation May Occur on a Data Page-Fault
X20		X	No Fix	CPUID Leaf 0x80000006 May Provide the Incorrect Value for an 8-Way Associative Cache
X21	X		PlanFix	Snoops during the Execution of a HLT (Halt) Instruction May Lead to Unexpected System Behavior

NO.	B1	C0	Plans	ERRATA
X22	X	X	No Fix	Invalid Entries in Page-Directory-Pointer-Table-Register (PDPTR) May Cause General Protection (#GP) Exception if the Reserved Bits are Set to One
X23	X	X	No Fix	INIT Does Not Clear Global Entries in the TLB
X24	X	X	No Fix	Use of Memory Aliasing with Inconsistent Memory Type May Cause System Hang
X25	X	X	No Fix	Machine Check Exception May Occur When Interleaving Code between Different Memory Types
X26	X	X	No Fix	Split I/O Writes Adjacent to Retry of APIC End of Interrupt (EOI) Request May Cause Livelock Condition
X27	X	X	No Fix	General Protection (#GP) Fault May Not Be Signaled On Data Segment Limit Violation above 4G Limit
X28	X	X	No Fix	DR3 Address Match on MOVD/MOVQ/MOVNTQ Memory Store Instruction May Incorrectly Increment Performance Monitoring Count for Saturating SIMD Instructions Executed (Event B1h)
X29	X	X	No Fix	Pending x87 FPU Exceptions (#MF) Following STI May Be Serviced before Higher Priority Interrupts
X30	X	X	No Fix	Processor INIT# Will Cause a System Hang If Triggered during an NMI Interrupt Routine Performed during Shutdown
X31	X	X	No Fix	Certain Performance Monitoring Counters Related to Bus, L2 Cache and Power Management Are Inaccurate
X32	X	X	No Fix	CS Limit Violation on RSM May Be Serviced before Higher Priority Interrupts/Exceptions
X33	X	X	No Fix	A Write to an APIC Register Sometimes May Appear to Have Not Occurred
X34	X	X	No Fix	The Processor May Report a #TS Instead of a #GP Fault
X35	X	X	No Fix	BTS Message May Be Lost When the STPCLK# Signal Is Active
X36	X	X	No Fix	Last Exception Record (LER) MSRs May Be Incorrectly Updated
X37	X	X	No Fix	Writing the Local Vector Table (LVT) when an Interrupt is Pending May Cause an Unexpected Interrupt
X38	X	X	No Fix	Global Pages in the Data Translation Look-Aside Buffer (DTLB) May Not Be Flushed by RSM Iinstruction before Restoring the Architectural State from SMRAM
X39	X	X	No Fix	Using 2M/4M pages When A20M# Is Asserted May Result in Incorrect Address Translations
X40	X	X	No Fix	Premature Execution of a Load Operation Prior to Exception Handler Invocation
X41	X	X	No Fix	#GP Fault Is NOT Generated on Writing IA32_MISC_ENABLE [34] When Execute Disable (XD) Is Not Supported
X42	X	X	No Fix	SSE/SSE2 Streaming Store Resulting in a Self-Modifying Code (SMC) Event May Cause Unexpected Behavior
X43	X	X	No Fix	Incorrect Address Computed For Last Byte of FXSAVE/FXRSTOR Image Leads to Partial Memory Update

NO.	B1	C0	Plans	ERRATA
X44	X	X	No Fix	Values for LBR/BTS/BTM Will Be Incorrect after an Exit from SMM
X45	X	X	No Fix	The BS Flag in DR6 May Be Set for Non-Single-Step #DB Exception
X46	X	X	No Fix	BTM/BTS Branch-From Instruction Address May Be Incorrect for Software Interrupts
X47	X	X	No Fix	Fault on ENTER Instruction May Result in Unexpected Values on Stack Frame
X48	X	X	No Fix	Unaligned Accesses to Paging Structures May Cause the Processor to Hang
X49	X	X	No Fix	INVLPG Operation for Large (2M/4M) Pages May be Incomplete under Certain Conditions
X50	X	X	No Fix	Page Access Bit May be Set Prior to Signaling a Code Segment Limit Fault
X51	X	X	No Fix	EFLAGS, CR0, CR4 and the EXF4 Signal May Be Incorrect after Shutdown
X52	X	X	No Fix	Store Ordering May Be Incorrect between WC and WP Memory Types

NO.	SPECIFICATION CLARIFICATION
X1	Specification Clarification with Respect to Time-stamp Counter – Removed
X2	Thermal Diode Offset Specification Clarification

Number	SPECIFICATION CHANGES
X1	AGTL+ Buffer On Resistance (Ron) Specification Correction

Number	DOCUMENTATION CHANGES
	There are no Documentation Changes in this Specification Update revision.

§

Identification Information

The Intel Pentium M processor on 90 nm process with 2-MB L2 cache can be identified by the following values:

Family[1]	Model[2]	Brand ID[3]
0110	1101	00010110

NOTES:

1. The Family corresponds to bits [11:8] of the EDX register after Reset, bits [11:8] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register.
2. The Model corresponds to bits [7:4] of the EDX register after Reset, bits [7:4] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register.
3. The Brand ID corresponds to bits [7:0] of the EBX register after the CPUID instruction is executed with a1 in the EAX register.

Table 1. Identification Table

QDF / S-spec	Processor Numbers	Product Stepping	FSB Frequency	CPU Signature	Core Speed		Package Micro-FCBGA-Pb= µ-BGA Lead Free	QDF Notes
					Highest Freq. Mode (HFM)	Lowest Freq. Mode (LFM)		
SL86G	730	C-0	533	06D8h	1.6 GHz	800 MHz	Micro-FCPGA	2,7
SL7SA	740	C-0	533	06D8h	1.73 GHz	800 MHz	Micro-FCPGA	2,7
SL7S9	750	C-0	533	06D8h	1.86 GHz	800 MHz	Micro-FCPGA	2,7
SL7SM	760	C-0	533	06D8h	2.0 GHz	800 MHz	Micro-FCPGA	2,7
SL7SL	770	C-0	533	06D8h	2.13 GHz	800 MHz	Micro-FCPGA	2,8
SL7VB	780	C-0	533	06D8h	2.26 GHz	800 MHz	Micro-FCPGA	9,2
SL86M	730	C-0	533	06D8h	1.6 GHz	800 MHz	Micro-FCBGA	2,7
SL7S8	740	C-0	533	06D8h	1.73 GHz	800 MHz	Micro-FCBGA	2,7
SL7SR	750	C-0	533	06D8h	1.86 GHz	800 MHz	Micro-FCBGA	2,7
SL7SQ	760	C-0	533	06D8h	2.0 GHz	800 MHz	Micro-FCBGA	2,7
SL7SP	770	C-0	533	06D8h	2.13 GHz	800 MHz	Micro-FCBGA	2,8
SL7SN	780	C-0	533	06D8h	2.26 GHz	800 MHz	Micro-FCBGA	9,2
SL86B	740	C-0	533	06D8h	1.73 GHz	800 MHz	Micro-FCBGA-Pb	2,7
SL86A	750	C-0	533	06D8h	1.86 GHz	800 MHz	Micro-FCBGA-Pb	2,7
SL869	760	C-0	533	06D8h	2.0 GHz	800 MHz	Micro-FCBGA-Pb	2,7
SL868	770	C-0	533	06D8h	2.13 GHz	800 MHz	Micro-FCBGA-Pb	2,8
SL8QK	780	C-0	533	06D8h	2.26 GHz	800 MHz	Micro-FCBGA-Pb	9,2
SL8QF	778	C-0	400	06D8h	1.6 GHz	600 MHz	Micro-FCBGA	2,3
SL89X	758	C-0	400	06D8h	1.5 GHz	600 MHz	Micro-FCBGA	2,3
SL8A3	723	C-0	400	06D8h	1.0 GHz	600 MHz	Micro-FCBGA	4,5
SL8LM	733J	C-0	400	06D8h	1.1 GHz	600 MHz	Micro-FCBGA	4,10
SL8A2	733J	C-0	400	06D8h	1.1 GHz	600 MHz	Micro-FCBGA	4,5
SL89Z	753	C-0	400	06D8h	1.2 GHz	600 MHz	Micro-FCBGA	4,5
SL8LL	753	C-0	400	06D8h	1.2 GHz	600 MHz	Micro-FCBGA	4,10
SL8QG	778	C-0	400	06D8h	1.6 GHz	600 MHz	Micro-FCBGA-Pb	2,3
SL89M	758	C-0	400	06D8h	1.5 GHz	600 MHz	Micro-FCBGA-Pb	2,3
SL89R	723	C-0	400	06D8h	1.0 GHz	600 MHz	Micro-FCBGA-Pb	2,3
SL8LT	733J	C-0	400	06D8h	1.1 GHz	600 MHz	Micro-FCBGA-Pb	4,10
SL89Q	733J	C-0	400	06D8h	1.1 GHz	600 MHz	Micro-FCBGA-Pb	4,5
SL89P	753	C-0	400	06D8h	1.2 GHz	600 MHz	Micro-FCBGA-Pb	4,5

QDF / S-spec	Processor Numbers	Product Stepping	FSB Frequency	CPU Signature	Core Speed		Package Micro-FCBGA-Pb= µ-BGA Lead Free	QDF Notes
					Highest Freq. Mode (HFM)	Lowest Freq. Mode (LFM)		
SL8LS	753	C-0	400	06D8h	1.2 GHz	600 MHz	Micro-FCBGA-Pb	4,10
SL8LK	773	C-0	400	06D8h	1.3 GHz	600 MHz	Micro-FCBGA	4,10
SL8LR	773	C-0	400	06D8h	1.3 GHz	600 MHz	Micro-FCBGA-Pb	4,10
SL8U8	745	C-0	400	06D8h	1.8 GHz	600 MHz	Micro-FCBGA-Pb	2,11,12
SL8U6	745	C-0	400	06D8h	1.8 GHz	600 MHz	Micro-FCPGA	2,11,12
SL8TV	738	C-0	400	06D8h	1.4 GHz	600 MHz	Micro-FCBGA-Pb	2,3,12
SL89N	738	C-0	400	06D8h	1.4 GHz	600 MHz	Micro-FCBGA-Pb	2,3,12
SL8LM	733	C-0	400	06D8h	1.1 GHz	600 MHz	Micro-FCBGA	2,3,12
SL89Y	738	C-0	400	06D8h	1.4 GHz	600 MHz	Micro-FCPGA	2,3
SL7EM	755	B-1	400	06D6h	2.0 GHz	600 MHz	Micro-FCPGA	1,2
SL7EL	755	B-1	400	06D6h	2.0 GHz	600 MHz	Micro-FCBGA	1,2
SL7EN	745	B-1	400	06D6h	1.8 GHz	600 MHz	Micro-FCPGA	1,2
SL7EQ	745	B-1	400	06D6h	1.8 GHz	600 MHz	Micro-FCBGA	1,2
SL7EP	735	B-1	400	06D6h	1.7 GHz	600 MHz	Micro-FCPGA	1,2
SL7ER	735	B-1	400	06D6h	1.7 GHz	600 MHz	Micro-FCBGA	1,2
SL7EG	725	B-1	400	06D6h	1.6 GHz	600 MHz	Micro-FCPGA	1,2
SL7F2	725	B-1	400	06D6h	1.6 GHz	600 MHz	Micro-FCBGA	1,2
SL7GL	715	B-1	400	06D6h	1.5 GHz	600 MHz	Micro-FCPGA	1,2
SL7GK	715	B-1	400	06D6h	1.5 GHz	600 MHz	Micro-FCBGA	1,2
SL7F3	738	B-1	400	06D6h	1.4 GHz	600 MHz	Micro-FCBGA	2,3
SL7F4	733	B-1	400	06D6h	1.1 GHz	600 MHz	Micro-FCBGA	4,5
SL7VD	733	B-1	400	06D6h	1.1 GHz	600 MHz	Micro-FCBGA	4,5
SL7V2	723	B-1	400	06D6h	1.0 GHz	600 MHz	Micro-FCBGA	4,5

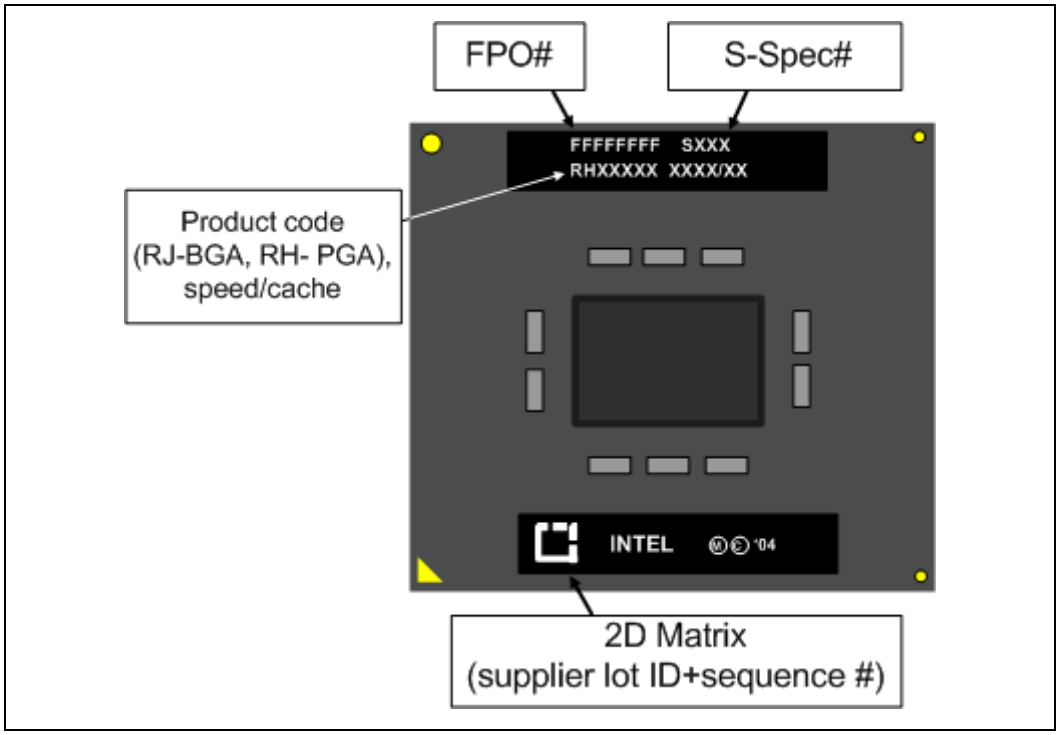
NOTES:

1. Multiple HFM VID's; VCC_CORE = 1.340-1.276 V for Highest Frequency Mode (HFM).
2. VCC_CORE = 0.988 V for Lowest Frequency Mode (LFM).
3. VCC_CORE = 1.116 V for Highest Frequency Mode (HFM).
4. VCC_CORE = 0.812 V for Lowest Frequency Mode (LFM).
5. VCC_CORE = 0.940 V for Highest Frequency Mode (HFM).
6. Multiple HFM VID's; VCC_CORE = 1.356-1.308 V for Highest Frequency Mode (HFM)
7. VCC_CORE = 1.260-1.356 V for Highest Frequency Mode (HFM).
8. VCC_CORE = 1.260-1.372 V for Highest Frequency Mode (HFM).
9. VCC_CORE = 1.260-1.404 V for Highest Frequency Mode (HFM).
10. VCC_CORE = 0.876-0.956 V for Highest Frequency Mode (HFM).
11. VCC_CORE = 1.276-1.340 V for Highest Frequency Mode (HFM).
12. This part is for embedded customers

Component Marking Information

The Intel Pentium M Processor on 90 nm process with 2-MB L2 cache may be identified by the following component markings:

Figure 1. Intel Pentium M Processor on 90 nm Process with 2-MB L2 Cache (Micro-FCPGA/FCBGA) Markings



§

Errata

X1. Code Segment (CS) Is Wrong on SMM Handler When SMBASE Is Not Aligned

Problem: With SMBASE being relocated to a non-aligned address, during SMM entry the CS can be improperly updated, which can lead to an incorrect SMM handler.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially available software or system.

Workaround: Align SMBASE to 32 KB.

Status: For the steppings affected, see the Summary of Tables of Changes.

X2. IFU/BSU Deadlock May Cause System Hang

Problem: A lockable instruction with memory operand that spans across two pages may, given some rare internal conditions, hang the system.

Implication: When this erratum occurs, the system may hang. Intel has not observed this erratum with any commercially available software or system.

Workaround: Lockable data should always be contained in a single page.

Status: For the steppings affected, see the Summary of Tables of Changes.

X3. Memory Aliasing with Inconsistent A and D Bits May Cause Processor Deadlock

Problem: In the event that software implements memory aliasing by having two page directory entries (PDEs) point to a common page table entry (PTE) and the Accessed and Dirty bits for the two PDEs are allowed to become inconsistent the processor may become deadlocked.

Implication: This erratum has not been observed with commercially available software.

Workaround: Software that needs to implement memory aliasing in this way should manage the consistency of the Accessed and Dirty bits.

Status: For the steppings affected, see the Summary of Tables of Changes.

X4. RDMSR or WRMSR to Invalid MSR Address May Not Cause GP Fault

Problem: The RDMSR and WRMSR instructions allow reading or writing of MSR's (Model Specific Registers) based on the index number placed in ECX. The processor should reject access to any reserved or unimplemented MSRs by generating #GP(0). However, there are some invalid MSR addressers for which the processor will not generate #GP(0). This erratum has not been observed with commercially available software.

Implication: For RDMSR, undefined values will be read into EDX:EAX. For WRMSR, undefined processor behavior may result.

Workaround: Do not use invalid MSR addresses with RDMSR or WRMSR.

Status: For the steppings affected, see the Summary of Tables of Changes.

X5. Unable to Disable Reads/Writes to Performance Monitoring Related MSRs

Problem: The Performance Monitoring Available bit in the miscellaneous processor features MSR (IA32_MISC_ENABLES.7) was defined so that when it is cleared to a 0, RDMSR/ WRMSR/RDPMC instructions would return all zeros for reads of and prevent any write to Performance Monitoring related MSRs. Currently it is possible to read from or write to Performance Monitoring related MSRs when the Performance Monitoring Available bit is cleared to a 0.

Implication: It is not possible to disallow reads and writes to the Performance Monitoring MSRs. Intel has not observed this erratum with commercially available software of system.

Workaround: None.

Status: For the steppings affected, see the Summary of Tables of Changes.

X6. Move to Control Register Instruction May Generate a Breakpoint Report

Problem: A move (MOV) to Control register (CR) instruction where Control register is CR0, CR3 or CR4 may generate a breakpoint report.

Implication: MOV to Control Register Instruction is not expected to generate a breakpoint report.

Workaround: Ignore breakpoint data from MOV to CR instruction.

Status: For the steppings affected, see the Summary of Tables of Changes.

X7. Error in Instruction Fetch Unit (IFU) Can Result in an Erroneous Machine Check-Exception (#MC)

Problem: A rare combination of events including the generation of a bus lock(s), the execution of a WBINVD instruction, and a page accessed or dirty bit assist may result in an erroneous Machine Check-Exception (#MC).

Implication: Due to this erratum, unexpected machine check-exception (#MC) is generated. Intel has not been able to reproduce this erratum with commercially available software.

Workaround: It is possible for BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary of Tables of Changes.

X8. Code Fetch Matching Disabled Debug Register May Cause Debug Exception

Problem: The bits L0-3 and G0-3 enable breakpoints local to a task and global to all tasks, respectively. If one of these bits is set, a breakpoint is enabled, corresponding to the addresses in the debug registers DR0-DR3. If at least one of these breakpoints is enabled, any of these registers are disabled (i.e., Ln and Gn are 0), and RWn for the disabled register is 00 (indicating a breakpoint on instruction execution), normally an instruction fetch will not cause an instruction-breakpoint fault based on a match with the address in the disabled register(s). However, if the address in a disabled register matches the address of a code fetch which also results in a page fault, an instruction-breakpoint fault will occur.

Implication: While debugging software, extraneous instruction-breakpoint faults may be encountered if breakpoint registers are not cleared when they are disabled. Debug software which does not implement a code breakpoint handler will fail, if this occurs. If a handler is present, the fault will be serviced. Mixing data and code may exacerbate this problem by allowing disabled data breakpoint registers to break on an instruction fetch.

Workaround: The debug handler should clear breakpoint registers before they become disabled.

Status: For the steppings affected, see the Summary of Tables of Changes.

X9. Upper Four PAT Entries Not Usable with Mode B or Mode C Paging

Problem: The Page Attribute Table (PAT) contains eight entries, which must all be initialized and considered when setting up memory types for the Pentium M processor. However, in Mode B or Mode C paging, the upper four entries do not function correctly for 4-Kbyte pages. Specifically, bit 7 of page table entries that translate addresses to 4-Kbyte pages should be used as the upper bit of a 3-bit index to determine the PAT entry that specifies the memory type for the page. When Mode B (CR4.PSE = 1) and/or Mode C (CR4.PAE) are enabled, the processor forces this bit to zero when determining the memory type regardless of the value in the page table entry. The upper four entries of the PAT function correctly for 2-Mbyte and 4-Mbyte large pages (specified by bit 12 of the page directory entry for those translations).

Implication: Only the lower four PAT entries are useful for 4-KB translations when Mode B or C paging is used. In Mode A paging (4-Kbyte pages only), all eight entries may be used. All eight entries may be used for large pages in Mode B or C paging.

Workaround: None.

Status: For the steppings affected, see the Summary of Tables of Changes.

X10. SSE/SSE2 Streaming Store Resulting in a Self-Modifying Code (SMC) Event May Cause Unexpected Behavior

Problem: An SSE or SSE2 streaming store that results in a Self-Modifying Code (SMC) event may cause unexpected behavior. The SMC event occurs on a full address match of code contained in L1 cache.

Implication: Due to this erratum, any of the following events may occur:

1. A data access break point may be incorrectly reported on the instruction pointer (IP) just before the store instruction.
2. A non-cacheable store can appear twice on the external bus (the first time it will write only 8 bytes, the second time it will write the entire 16 bytes).

Intel has not observed this erratum with any commercially available software. This erratum has been seen in a synthetic test environment.

Workaround: None.

Status: For the steppings affected, see the Summary of Tables of Changes.

X11. Code Segment Limit Violation May Occur on 4 Gigabyte Limit Check

Problem: Code Segment limit violation may occur on 4 Gigabyte limit check when the code stream wraps around in a way that one instruction ends at the last byte of the segment and the next instruction begins at 0x0.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially available software, or system.

Workaround: Avoid code that wraps around segment limit.

Status: For the steppings affected, see the Summary of Tables of Changes.

X12. FST Instruction with Numeric and Null Segment Exceptions May Cause General Protection Faults to Be Missed and FP Linear Address (FLA) Mismatch

Problem: FST instruction combined with numeric and null segment exceptions may cause General Protection Faults to be missed and FP Linear Address (FLA) mismatch.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially available software, or system.

Workaround: None identified.

Status: For the steppings affected, see the Summary of Tables of Changes.

X13. Removed; See Erratum X1.

X14. Page with PAT (Page Attribute Table) Set to USWC (Uncacheable Speculative Write Combine) While Associated MTRR (Memory Type Range Register) Is UC (Uncacheable) May Consolidate to UC

Problem: A page whose PAT memory type is USWC while the relevant MTRR memory type is UC, the consolidated memory type may be treated as UC (rather than WC as specified in *IA-32 Intel® Architecture Software Developer's Manua*l).

Implication: When this erratum occurs, the memory page may be as UC (rather than WC). This may have a negative performance impact.

Workaround: None identified.

Status: For the steppings affected, see the Summary of Tables of Changes.

X15. Under Certain Conditions LTR (Load Task Register) Instruction May Result in System Hang

Problem: An LTR instruction may result in a system hang if all the following conditions are met:

1. Invalid data selector of the TR (Task Register) resulting with either #GP (General Protection Fault) or #NP (Segment Not Present Fault).
2. GDT (Global Descriptor Table) is not 8-bytes aligned.
3. Data BP (breakpoint) is set on cache line containing the descriptor data.

Implication: This erratum may result in system hang if all conditions have been met. This erratum has not been observed in commercial operating systems or software. For performance reasons, GDT is typically aligned to 8-bytes.

Workaround: Software should align GDT to 8-bytes.

Status: For the steppings affected, see the Summary of Tables of Changes.

X16. Loading from Memory Type USWC (Uncacheable Speculative Write Combine) May Get Its Data Internally Forwarded from a Previous Pending Store

Problem: A load from memory type USWC may get its data internally forwarded from a pending store. As a result, the expected load may never be issued to the external bus.

Implication: When this erratum occurs, a USWC load request may be satisfied without being observed on the external bus. There are no known usage models where this behavior results in any negative side-effects.

Workaround: Do not use memory type USWC for memory that has read side-effects.

Status: For the steppings affected, see the Summary of Table of Changes.

X17. FXSAVE after FNINIT without an Intervening FP (Floating Point) Instruction May Save Uninitialized Values for FDP (x87 FPU Instruction Operand (Data) Pointer Offset) and FDS (x87 FPU Instruction Operand (Data) Pointer Selector)

Problem: An FXSAVE after FNINIT without an intervening FP instruction may save uninitialized values for FDP and FDS.

Implication: When this erratum occurs, the values for FDP/FDS in the FXSAVE structure may appear to be random values. These values will be initialized by the first FP instruction executed after the FXRSTOR that restore the saved floating point state. Any FP instruction with memory operand will initialize FDP/FDS. Intel has not observed this erratum with any commercially available software.

Workaround: After an FINIT, do not expect the FXSAVE memory image to be correct, until at least one FP instruction with a memory operand has been executed.

Status: For the steppings affected, see the Summary of Tables of Changes.

X18. FSTP (Floating Point Store) Instruction under Certain Conditions May Result In Erroneously Setting a Valid Bit on an FP (Floating Point) Stack Register

Problem: An FSTP instruction with a PDE/PTE (Page Directory Entry/Page Table Entry) A/D bit update followed by user mode access fault due to a code fetch to a page that has supervisor only access permission may result in erroneously setting a valid bit of an FP stack register. The FP top of stack pointer is unchanged.

Implication: This erratum may cause an unexpected stack overflow.

Workaround: User mode code should not count on being able to recover from illegal accesses to memory regions protected with supervisor only access when using FP instructions.

Status: For the steppings affected, see the Summary of Tables of Changes.

X19. An Execute Disable Bit Violation May Occur on a Data Page-Fault

Problem: Under a combination of internal events, unexpected Execute Disable violations may occur on data accesses that are Execute Disable protected.

Implication: This erratum may cause unexpected Execute Disable violations.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary of Tables of Changes.

X20. CPUID Leaf 0x80000006 May Provide the Incorrect Value for an 8-Way Associative Cache

Problem: CPUID leaf 0x80000006 may return 0x8 in ECX [15:12] to indicate 8-way associative cache, but the correct encoding for an 8-way associative cache is 0x6.

Implication: Software that depends on the associativity of the cache may not function correctly.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary of Tables of Changes.

X21. Snoops during the Execution of a HLT (Halt) Instruction May Lead to Unexpected System Behavior

Problem: If during the execution of a HLT instruction an external snoop causes an eviction from the instruction fetch unit (IFU) instruction cache, the processor may, on exit from the HLT state, erroneously read stale data from the victim cache.

Implication: This erratum may lead to unexpected system behavior. Intel has only observed this condition in non-mobile configurations.

Workaround: It is possible for the BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the Summary of Tables of Changes.

X22. Invalid Entries in Page-Directory-Pointer-Table-Register (PDPTR) May Cause General Protection (#GP) Exception If the Reserved Bits are Set to One

Problem: Invalid entries in Page-Directory-Pointer-Table-Register (PDPTR) that have the reserved bits set to one, may cause a General Protection (#GP) exception.

Implication: Intel has not observed this erratum with any commercially available software.

Workaround: Do not set the reserved bits to one when PDPTR entries are invalid.

Status: For the steppings affected, see the Summary of Table of Changes.

X23. INIT Does Not Clear Global Entries in the TLB

Problem: INIT may not flush a TLB entry when:

1. The processor is in protected mode with paging enabled and the page global enable flag is set (PGE bit of CR4 register)
2. G bit for the page table entry is set
3. TLB entry is present in TLB when INIT occurs

Implication: Software may encounter unexpected page fault or incorrect address translation due to a TLB entry erroneously left in TLB after INIT.

Workaround: Write to CR3, CR4 (setting bits PSE, PGE or PAE) or CR0 (setting bits PG or PE) registers before writing to memory early in BIOS code to clear all the global entries from TLB.

Status: For the steppings affected, see the Summary of Tables of Changes.

X24. Use of Memory Aliasing with Inconsistent Memory Type May Cause System Hang

Problem: Software that implements memory aliasing by having more than one linear addresses mapped to the same physical page with different cache types may cause the system to hang. This would occur if one of the addresses is non-cacheable used in code segment and the other a cacheable address. If the cacheable address finds its way in instruction cache, and non-cacheable address is fetched in IFU, the processor may invalidate the non-cacheable address from the fetch unit. Any micro-architectural event that causes instruction restart will expect this instruction to still be in fetch unit and lack of it will cause system hang.

Implication: This erratum has not been observed with commercially available software.

Workaround: Although it is possible to have a single physical page mapped by two different linear addresses with different memory types, Intel has strongly discouraged this practice as it may lead to undefined results. Software that needs to implement memory aliasing should manage the memory type consistency.

Status: For the steppings affected, see the Summary of Tables of Changes.

X25. Machine Check Exception May Occur When Interleaving Code between Different Memory Types

Problem: A small window of opportunity exists where code fetches interleaved between different memory types may cause a machine check exception. A complex set of micro-architectural boundary conditions is required to expose this window.

Implication: Interleaved instruction fetches between different memory types may result in a machine check exception. The system may hang if machine check exceptions are disabled. Intel has not observed the occurrence of this erratum while running commercially available applications or operating systems.

Workaround: Software can avoid this erratum by placing a serializing instruction between code fetches between different memory types.

Status: For the steppings affected, see the Summary of Tables of Changes.

X26. Split I/O Writes Adjacent to Retry of APIC End of Interrupt (EOI) Request May Cause Livelock Condition

Problem: When Split I/O instruction writes occur adjacent to a retry of a Local APIC End of Interrupt (EOI) request by the chipset, a livelock condition may result. The required sequences of events are:

1. The processor issues a Local APIC EOI message.
2. The chipset responds with a retry because its downstream ports are full. It expects the processor to return with the same EOI request.
3. The processor issues a Split I/O write instruction instead.
4. The chipset responds with a retry because it expected the APIC EOI.
5. The processor insists the Split I/O write instruction must be completed and issues write instruction again.

Implication: A processor livelock may occur causing a system hang. This issue has only been observed in synthetic lab testing conditions and has not been seen in any commercially available applications. The erratum does not occur with Intel mobile chipset based platforms.

Workaround: Use the PIC instead of the APIC for the interrupt controller.

Status: For the steppings affected, see the Summary of Tables of Changes.

X27. General Protection (#GP) Fault May Not Be Signaled on Data Segment Limit Violation above 4-G Limit

Problem: Memory accesses to flat data segments (base = 00000000h) that occur above the 4-G limit (0ffffffffh) may not signal a #GP fault.

Implication: When such memory accesses occur, the system may not issue a #GP fault.

Workaround: Software should ensure that memory accesses do not occur above the 4-G limit (0ffffffffh).

Status: For the steppings affected, see the Summary of Tables of Changes.

X28. DR3 Address Match on MOVD/MOVQ/MOVNTQ Memory Store Instruction May Incorrectly Increment Performance Monitoring Count for Saturating SIMD Instructions Executed (Event B1h)

Problem: Performance monitoring for Event B1h normally increments on saturating SIMD instruction executed. Regardless of DR7 programming, if the linear address of a memory store MOVD/MOVQ/MOVNTQ instruction executed matches the address in DR3, the B1h counter may be incorrectly incremented.

Problem: The value observed for performance monitoring count for saturating SIMD instructions executed may be too high.

Workaround: None identified.

Status: For the steppings affected, see the Summary of Tables of Changes.

X29. Pending x87 FPU Exceptions (#MF) Following STI May Be Serviced before Higher Priority Interrupts

Problem: Interrupts that are pending prior to the execution of the STI (Set Interrupt Flag) instruction are serviced immediately after the STI instruction is executed. Because of this erratum, if following STI, an instruction that triggers a #MF is executed while STPCLK#, Enhanced Intel SpeedStep® Technology transitions or Thermal Monitor 1 events occur, the pending #MF may be serviced before higher priority interrupts.

Implication: Software may observe #MF being serviced before higher priority interrupts.

Workaround: None identified.

Status: For the steppings affected, see the Summary of Tables of Changes.

X30. Processor INIT# Will Cause a System Hang if Triggered during an NMI Interrupt Routine Performed during Shutdown

Problem: During the execution of an NMI interrupt handler, if shutdown occurs followed by the INIT# signal being triggered, the processor will attempt initialization but fail soft reset.

Implication: Due to this erratum the system may hang.

Workaround: None identified.

Status: For the steppings affected, see the Summary of Tables of Changes.

X31. Certain Performance Monitoring Counters Related to Bus, L2 Cache and Power Management Are Inaccurate

Problem: All Performance Monitoring Counters in the ranges 21H-3DH and 60H-7FH may have inaccurate results up to ±7.

Implication: There may be a small error in the affected counts.

Workaround: None identified.

Status: For the steppings affected, see the Summary of Tables of Changes.

X32. CS Limit Violation on RSM May Be Serviced before Higher Priority Interrupts/Exceptions

Problem: When the processor encounters a CS (Code Segment) limit violation, a #GP (General Protection Exception) fault is generated after all higher priority Interrupts and exceptions are serviced. Because of this erratum, if RSM (Resume from System Management Mode) returns to execution flow where a CS limit violation occurs, the #GP fault may be serviced before a higher priority Interrupt or Exception (e.g. NMI (Non-Maskable Interrupt), Debug break(#DB), Machine Check (#MC), etc.).

Implication: Operating systems may observe a #GP fault being serviced before higher priority Interrupts and Exceptions.

Workaround: None identified.

Status: For the steppings affected, see the Summary of Tables of Changes.

X33. A Write to an APIC Register Sometimes May Appear to Have Not Occurred

Problem: With respect to the retirement of instructions, stores to the uncacheable memory-based APIC register space are handled in a non-synchronized way. For example if an instruction that masks the interrupt flag, e.g. CLI, is executed soon after an uncacheable write to the Task Priority Register (TPR) that lowers the APIC priority, the interrupt masking operation may take effect before the actual priority has been lowered. This may cause interrupts whose priority is lower than the initial TPR, but higher than the final TPR, to not be serviced until the interrupt enabled flag is finally set, i.e. by STI instruction. Interrupts will remain pending and are not lost.

Implication: In this example the processor may allow interrupts to be accepted but may delay their service.

Workaround: This non-synchronization can be avoided by issuing an APIC register read after the APIC register write. This will force the store to the APIC register before any subsequent instructions are executed. No commercial operating system is known to be impacted by this erratum.

Status: For the steppings affected, see the Summary of Tables of Changes.

X34. The Processor May Report a #TS Instead of a #GP Fault

Problem: A jump to a busy TSS (Task-State Segment) may cause a #TS (invalid TSS exception) instead of a #GP fault (general protection exception).

Implication: Operation systems that access a busy TSS may get invalid TSS fault instead of a #GP fault. Intel has not observed this erratum with any commercially available software.

Workaround: None identified.

Status: For the steppings affected, see the Summary of Tables of Changes.

X35. BTS Message May Be Lost When the STPCLK# Signal Is Active

Implication: STPCLK# is asserted to enable the processor to enter a low-power state. Under some circumstances, when STPCLK# becomes active, a pending BTS (Branch Trace Store) message may be either lost and not written or written with corrupted branch address to the Debug Store area.

Implication: BTS messages may be lost in the presence of STPCLK# assertions.

Workaround: None identified.

Status: For the steppings affected, see the Summary of Tables of Changes.

X36. Last Exception Record (LER) MSRs May Be Incorrectly Updated

Problem: The LASTINTTOIP and LASTINTFROMIP MSRs (1DDH-1DEH) may contain incorrect values after the following events: masked SSE2 floating-point exception, StopClk, NMI and INT.

Implication: The value of the LER MSR may be incorrectly updated to point to a SIMD Floating-Point instruction even though no exception occurred on that instruction or to point to an instruction that was preceded by a StopClk interrupt or rarely not to be updated on Interrupts (NMI and INT).

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

X37. Writing the Local Vector Table (LVT) When an Interrupt Is Pending May Cause an Unexpected Interrupt

Problem: If a local interrupt is pending when the LVT entry is written, an interrupt may be taken on the new interrupt vector even if the mask bit is set.

Implication: An interrupt may immediately be generated with the new vector when a LVT entry is written, even if the new LVT entry has the mask bit set. If there is no Interrupt Service Routine (ISR) set up for that vector the system will GP fault. If the ISR does not do an End of Interrupt (EOI) the bit for the vector will be left set in the in-service register and mask all interrupts at the same or lower priority.

Workaround: Any vector programmed into an LVT entry must have an ISR associated with it, even if that vector was programmed as masked. This ISR routine must do an EOI to clear any unexpected interrupts that may occur. The ISR associated with the spurious vector does not generate an EOI, therefore the spurious vector should not be used when writing the LVT.

Status: For the steppings affected, see the Summary Tables of Changes.

X38. Global Pages in the Data Translation Look-Aside Buffer (DTLB) May Not Be Flushed by RSM instruction before Restoring the Architectural State from SMRAM

Problem: The Resume from System Management Mode (RSM) instruction does not flush global pages from the Data Translation Look-Aside Buffer (DTLB) prior to reloading the saved architectural state.

Implication: If SMM turns on paging with global paging enabled and then maps any of linear addresses of SMRAM using global pages, RSM load may load data from the wrong location.

Workaround: Do not use global pages in system management mode.

Status: For affected steppings see the Summary Table of Changes.

X39. Using 2M/4M Pages When A20M# Is Asserted May Result in Incorrect Address Translations

Problem: An external A20M# pin if enabled forces address bit 20 to be masked (forced to zero) to emulates real-address mode address wraparound at 1 megabyte. However, if all of the following conditions are met, address bit 20 may not be masked.

- paging is enabled
- a linear address has bit 20 set
- the address references a large page
- A20M# is enabled

Implication: When A20M# is enabled and an address references a large page the resulting translated physical address may be incorrect. This erratum has not been observed with any commercially available operating system.

Workaround: Operating systems should not allow A20M# to be enabled if the masking of address bit 20 could be applied to an address that references a large page. A20M# is normally only used with the first megabyte of memory.

Status: For affected steppings see the Summary Table of Changes.

X40. Premature Execution of a Load Operation Prior to Exception Handler Invocation

Problem: If any of the below circumstances occur it is possible that the load portion of the instruction will have executed before the exception handler is entered.

1. If an instruction that performs a memory load causes a code segment limit violation
2. If a waiting floating-point instruction or MMX instruction that performs a memory load has a floating-point exception pending
3. If an MMX or SSE instruction that performs a memory load and has either CR0.EM=1 (Emulation bit set), or a floating-point Top-of-Stack (FP TOS) not equal to 0, or a DNA exception pending

Implication: In normal code execution where the target of the load operation is to write back memory there is no impact from the load being prematurely executed, nor from the restart and subsequent re-execution of that instruction by the exception handler. If the target of the load is to uncached memory that has a system side-effect, restarting the instruction may cause unexpected system behavior due to the repetition of the side-effect.

Workaround: Code which performs loads from memory that has side-effects can effectively workaround this behavior by using simple integer-based load instructions when accessing side-effect memory and by ensuring that all code is written such that a code segment limit violation cannot occur as a part of reading from side-effect memory.

Status: For the steppings affected, see the Summary Tables of Changes.

X41. #GP Fault is NOT Generated on Writing IA32_MISC_ENABLE [34] When Execute Disable (XD) is Not Supported

Problem: #GP fault is not generated on writing to IA32_MISC_ENABLE [34] bit in a processor which does not support Execute Disable (XD) functionality.

Implication: Writing to IA32_MISC_ENABLE [34] bit is silently ignored without generating a fault.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

X42. SSE/SSE2 Streaming Store Resulting in a Self-Modifying Code (SMC) Event May Cause Unexpected Behavior

Problem: An SSE or SSE2 streaming store that results in a Self-Modifying Code (SMC) event may cause unexpected behavior. The SMC event occurs on a full address match of code contained in L1 cache.

Implication: Due to this erratum, any of the following events may occur:

1. A data access break point may be incorrectly reported on the instruction pointer (IP) just before the store instruction.
2. A non-cacheable store can appear twice on the external bus (the first time it will write only 8 bytes, the second time it will write the entire 16 bytes).

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

X43. Incorrect Address Computed for Last Byte of FXSAVE/FXRSTOR Image Leads to Partial Memory Update

Problem: A partial memory state save of the 512-byte FXSAVE image or a partial memory state restore of the FXRSTOR image may occur if a memory address exceeds the 64-KB limit while the processor is operating in 16-bit mode or if a memory address exceeds the 4GB limit while the processor is operating in 32-bit mode.

Implication: FXSAVE/FXRSTOR will incur a #GP fault due to the memory limit violation as expected but the memory state may be only partially saved or restored.

Workaround: Software should avoid memory accesses that wrap around the respective 16-bit and 32-bit mode memory limits.

Status: For affected steppings see the Summary Table of Changes.

X44. Values for LBR/BTS/BTM Will Be Incorrect after an Exit from SMM

Problem: After a return from SMM (System Management Mode), the CPU will incorrectly update the LBR (Last Branch Record) and the BTS (Branch Trace Store), hence rendering their data invalid. The corresponding data if sent out as a BTM on the system bus will also be incorrect. Note: This issue would only occur when one of the 3 above mentioned debug support facilities are used.

Implication: The value of the LBR, BTS, and BTM immediately after an RSM operation should not be used.

Workaround: None identified.

Status: For affected steppings see the Summary Table of Changes.

X45. The BS Flag in DR6 May Be Set for Non-Single-Step #DB Exception

Problem: DR6 BS (Single Step, bit 14) flag may be incorrectly set when the TF (Trap Flag, bit 8) of the EFLAGS Register is set, and a #DB (Debug Exception) occurs due to one of the following:

- DR7 GD (General Detect, bit 13) being bit set;
- INT1 instruction;
- Code breakpoint

Implication: The BS flag may be incorrectly set for non-single-step #DB exception.

Workaround: None identified.

Status: For the steppings affected, see the Summary Table of Changes.

X46. BTM/BTS Branch-From Instruction Address May Be Incorrect for Software Interrupts

Problem: When BTM (Branch Trace Message) or BTS (Branch Trace Store) is enabled, a software interrupt may result in the overwriting of BTM/BTS branch-from instruction address by the LBR (Last Branch Record) branch-from instruction address.

Implication: A BTM/BTS branch-from instruction address may get corrupted for software interrupts.

Workaround: None identified.

Status: For the steppings affected, see the Summary Table of Changes.

X47. Fault on ENTER Instruction May Result in Unexpected Values on Stack Frame

Problem: The ENTER instruction is used to create a procedure stack frame. Due to this erratum, if execution of the ENTER instruction results in a fault, the dynamic storage area of the resultant stack frame may contain unexpected values (i.e. residual stack data as a result of processing the fault).

Implication: Data in the created stack frame may be altered following a fault on the ENTER instruction. Please refer to "Procedure Calls For Block-Structured Languages" in the *IA-32 Intel® Architecture Software Developer's Manual, Vol. 1, Basic Architecture*, for information on the usage of the ENTER instructions. This erratum is not expected to occur in ring 3. Faults are usually processed in ring 0 and stack switch occurs when transferring to ring 0. Intel has not observed this erratum on any commercially-available software.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

X48. Unaligned Accesses to Paging Structures May Cause the Processor to Hang

Problem: When an unaligned access is performed on paging structure entries, accessing a portion of two different entries simultaneously, the processor may live lock.

Implication: When this erratum occurs, the processor may live lock causing a system hang.

Workaround: Do not perform unaligned accesses on paging structure entries.

Status: For the steppings affected, see the Summary Tables of Changes.

X49. INVLPG Operation for Large (2M/4M) Pages May Be Incomplete under Certain Conditions

Problem: The INVLPG instruction may not completely invalidate Translation Look-aside Buffer (TLB) entries for large pages (2M/4M) when both of the following conditions exist:

- Address range of the page being invalidated spans several Memory Type Range Registers (MTRRs) with different memory types specified
- INVLPG operation is preceded by a Page Assist Event (Page Fault (#PF) or an access that results in either A or D bits being set in a Page Table Entry (PTE))

Implication: Stale translations may remain valid in TLB after a PTE update resulting in unpredictable system behavior. Intel has not observed this erratum with any commercially available software.

Workaround: Software should ensure that the memory type specified in the MTRRs is the same for the entire address range of the large page.

Status: For the steppings affected, see the Summary Tables of Changes.

X50. Page Access Bit May Be Set Prior to Signaling a Code Segment Limit Fault

Problem: If code segment limit is set close to the end of a code page, then due to this erratum the memory page Access bit (A bit) may be set for the subsequent page prior to general protection fault on code segment limit.

Implication: When this erratum occurs, a non-accessed page which is present in memory and follows a page that contains the code segment limit may be tagged as accessed.

Workaround: Erratum can be avoided by placing a guard page (non-present or non-executable page) as the last page of the segment or after the page that includes the code segment limit.

Status: For the steppings affected, see the Summary Tables of Changes.

X51. EFLAGS, CR0, CR4 and the EXF4 Signal May Be Incorrect after Shutdown

Problem: When an unaligned access is performed on paging structure entries, accessing a portion of two different entries simultaneously, the processor may live lock.

Implication: When this erratum occurs, the processor may live lock causing a system hang.

Workaround: Do not perform unaligned accesses on paging structure entries.

Status: For the steppings affected, see the Summary Tables of Changes.

X52. Store Ordering May be Incorrect between WC and WP Memory Types

Problem: According to IA-32 Intel Architecture Software Developer's Manual, Volume 3A "Methods of Caching Available", WP (Write Protected) stores should drain the WC (Write Combining) buffers in the same way as UC (Uncacheable) memory type stores do. Due to this erratum, WP stores may not drain the WC buffers.

Implication: Memory ordering may be violated between WC and WP stores.

Workaround: None identified.

Status: For the steppings affected, see the Summary Tables of Changes.

§

Specification Changes

X1. AGTL+ Buffer on Resistance (R_{ON}) Specification Correction

The AGTL+ Buffer On Resistance (Ron) specification has been corrected for Intel Pentium M processor with 2-MB L2 cache and 533-MHz Front Side Bus. Intel® Pentium® M Processor with 2-MB L2 Cache and 533 MHz Front Side Bus Electrical, Mechanical, and Thermal Specification (EMTS) Revision 2.1 Table 3-7 AGTL+ Signal Group DC Specifications shows R_{ON} specification as:

Symbol	Parameter	Min	Typ	Max	Unit
R_{ON}	Buffer On Resistance	22	25	28	Ω

Should state:

Symbol	Parameter	Min	Typ	Max	Unit
R_{ON}	Buffer On Resistance	17.7	24.7	32.9	Ω

§

Specification Clarifications

X1. Removed

See the Revision History for details.

X2. Thermal Diode Offset Spec Clarification

The following text has been added to *Intel® Pentium® M Processor on 90-nm Process with 2-MB L2 Cache* and *Intel® Pentium® M Processor on 90-nm Process with 2-MB L2 Cache and 533 MHz Front Side Bus* datasheets chapter 5, section 5.1.2, Thermal Diode Offset:

The thermal diode offset model specific register, THERM_DIODE_OFFSET, is located at offset 03Fh accessed as a QWord. Bits 7:0 contain the thermal diode offset value in 0.5 °C resolution. This value should be subtracted from the diode measurement after the thermal diode ideality adjustments are made. Values from bits 7:0 of the MSR are interpreted as follows:

‘00000000’ = 0 °C
‘00000001’ = +0.5 °C
‘00000010’ = +1 °C
‘01111111’ = +63.5 °C
‘11111111’ = -0.5 °C
‘11111110’ = -1 °C
‘1000000’ = -64 °C

Appended to footnote of Table 5-7, *Thermal Diode Specification*:
Offset value is programmed in processor Model Specific Register 03Fh, “THERM_DIODE_OFFSET”

§

Documentation Changes

There are no Documentation Changes in this Specification Update revision.

§